Probabilistic Büchi Automata for LTL\GU

Dileep Kini and Mahesh Viswanathan

University of Illinois at Urbana-Champaign
Department of Computer Science

Abstract. LTL\$GU$ is a fragment of linear temporal logic (LTL), where negations appear only on propositions, and formulas are built using the temporal operators $\boldsymbol{X}$ (next), $\boldsymbol{F}$ (eventually), $\boldsymbol{G}$ (always), and $\boldsymbol{U}$ (until, with the restriction that no until operator occurs in the scope of an always operator. Our main result is the construction of probabilistic Büchi automata for this logic that are exponential in the size of the formula. One consequence of our construction is a new, improved EXPTIME model checking algorithm (as opposed to the previously known doubly exponential time) for Markov Decision Processes and LTL\$GU$ formulae.

1 Introduction

Starting with the seminal work of Vardi, Wolper, and Sistla [16], there has been a lot of interest in discovering efficient translations of Linear Temporal logic (LTL) formulae into small automata (see [15, 7, 9, 13, 12, 10, 2, 6] for example). The reason for this is that logic to automata translations have a direct impact on algorithms of verification and synthesis of systems [17]. When verifying systems, one is often satisfied with constructing nondeterministic Büchi automata for LTL formulae. However, for a couple of applications, general nondeterministic automata don't suffice — when synthesizing reactive modules for LTL specifications, deterministic automata are necessary, and when model checking Markov Decision Processes with respect to almost sure satisfaction of LTL specifications, one needs either deterministic or probabilistic automata. As a consequence, a series of papers recently present algorithms and tools for constructing deterministic automata from LTL specifications [9, 13, 12, 10, 11, 2, 6]; though in the worst case the size of the constructed automata are doubly exponential, these algorithms have been shown to construct small automata for a number of examples. In this paper, we investigate whether there are provable improvements in translating fragments of LTL to *probabilistic automata*, and explore whether these can then be exploited to improve the asymptotic complexity of the MDP model checking problem.

We consider the fragment LTL\GU, first introduced in [11]. In this logic, formulae are built from propositions and their negations using conjunction, disjunction, and the temporal operators $\boldsymbol{X}$ (next), $\boldsymbol{F}$(eventually), $\boldsymbol{G}$ (always), and $\boldsymbol{U}$ (until), with the restriction that no $\boldsymbol{U}$ operator appears in the scope of a $\boldsymbol{G}$ operator. We translate this logic into probabilistic Büchi automata (PBA) [3].

Probabilistic Büchi automata are like Büchi automata, except that they probabilistically choose the next state on reading an input symbol. On input w, such a machine can be seen as defining a probability measure on the space of all runs on w. A PBA is said to accept a (infinite length) string w iff the set of all accepting runs (i.e., runs that visit some final state infinitely often) have measure > 0.

Our main result is a translation of LTL$\backslash GU$ formulae into PBAs of exponential size. This should be contrasted with the observation that any translation from LTL$\backslash GU$ to deterministic automata must in the worst case result in automata that are doubly exponential in size [1]; in fact, this lower bound applies to any fragment of LTL that has $\vee$, $\wedge$, and $\boldsymbol{F}$. The main result of this paper also generalizes some of the results in [8] where exponential sized *weak probabilistic monitors* [1] are constructed for the LTL fragment with just the temporal operators $\boldsymbol{X}$ and $\boldsymbol{G}$.

We prove the main result by constructing an exponentially sized nondeterministic Büchi automaton that is *deterministic in the limit* for LTL$\backslash GU$ — an automaton is deterministic in the limit if the transitions from any state that is reachable from an accepting state are deterministic. We then use the observation that any assignment of non-zero probabilities to the nondeterministic choices of a limit deterministic NBA, results in a PBA that accepts the same language [3]. Our construction of limit deterministic automata for LTL$\backslash GU$ proceeds in two steps. First we construct limit deterministic automata for LTL(F,G) which is the LTL fragment without until, i.e., with just the temporal operators next, always, and eventually. Next, we observe that the automaton for $\varphi \in$ LTL$\backslash GU$ can be seen as the composition of two limit deterministic automata: one automata for the formula ψ, where all the until-free subformulae of φ are replaced by propositions, and another automaton for the until-free subformulae of φ. This composition is reminiscent of the master-slave composition in [6] and the composition of temporal testers [14] but with some differences.

Our construction of exponentially sized PBAs for LTL$\backslash GU$ has complexity theoretic consequences for model checking MDPs. Courcoubetis and Yannakakis [5] proved that the problem of model checking MDPs against LTL is 2EXPTIME-complete. Our automata construction, coupled with the algorithm outlined in [5], shows that model checking MDPs against LTL$\backslash GU$ is in EXPTIME; we prove a matching lower bound in this paper as well. Thus, for a large, expressively rich subclass of LTL specifications, our results provide an exponential improvement to the complexity of model checking MDPs.

The rest of the paper is organized as follows. In Section 2 we introduce the notations and definitions we use in the paper. In Section 3 we present a translation from LTL(F,G) to limit deterministic NBAs. In Section 4 we give a compositional style construction for formulae in LTL$\backslash GU$ by using the construction in the previous section. In Section 5 we reflect on the consequences of our results and finally give concluding remarks in Section 6.

[1] A weak finite state probabilistic monitor [4] is a PBA with the restriction that all states except a unique reject state are final, and all transitions from the unique rejecting state are self loops.

2 Preliminaries

First we introduce the notation we use throughout the paper. We use P to denote the set of propositions. An assignment ν is a function mapping all propositions to true or false. We use w to denote infinite words over a finite alphabet. We use $w[i]$ to denote the i^{th} symbol in the sequence w, and use w_i to denote the suffix $w[i]w[i+1]\dots$ of w starting at i. We use $[n]$ to denote all non-negative integers less than n that is $\{0, 1, \dots, n-1\}$. We shall use Σ, Γ to denote finite sets of symbols.

Definition 1 (Syntax). *The formulae in the fragment* LTL(F,G) *over* P *is given by the following syntax*

$$\varphi \quad ::= \quad p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \boldsymbol{X}\varphi \mid \boldsymbol{F}\varphi \mid \boldsymbol{G}\varphi \qquad p \in P$$

and the formulae in the fragment LTL$\backslash GU$ *are given by the synatx*

$$\psi \quad ::= \quad \varphi \mid \psi \wedge \psi \mid \psi \vee \psi \mid \boldsymbol{X}\psi \mid \psi \boldsymbol{U} \psi \qquad \varphi \in \text{LTL}(F,G)$$

Let $\mathbb{F}_\varphi$ and $\mathbb{G}_\varphi$ denote the set of all $\boldsymbol{F}$ and $\boldsymbol{G}$ subformulae of φ respectively. We drop the subscript whenever the formula φ is clear from the context. For a temporal operator $Op \in \{\boldsymbol{G}, \boldsymbol{F}, \boldsymbol{U}\}$ we use LTL(Op) to denote the fragment consisting of formulae built using $Op, \boldsymbol{X}, \vee, \wedge$ with negations appearing only on propositions.

Definition 2 (Semantics). *LTL formulae over a set* P *are interpreted over words* w *in* $(2^P)^\omega$. *The semantics of the logic are given by the following rules*

$$\begin{array}{lll@{\qquad}lll}
w \vDash p & \iff & p \in w[0] & w \vDash \boldsymbol{X}\varphi & \iff & w_1 \vDash \varphi \\
w \vDash \neg p & \iff & p \notin w[0] & w \vDash \boldsymbol{F}\varphi & \iff & \exists\, i : w_i \vDash \varphi \\
w \vDash \varphi \wedge \psi & \iff & w \vDash \varphi \text{ and } w \vDash \psi & w \vDash \boldsymbol{G}\varphi & \iff & \forall\, i : w_i \vDash \varphi \\
w \vDash \varphi \vee \psi & \iff & w \vDash \varphi \text{ or } w \vDash \psi & w \vDash \varphi \boldsymbol{U} \psi & \iff & \exists\, i : w_i \vDash \psi, \text{and} \\
 & & & & & \forall\, j < i : w_j \vDash \varphi
\end{array}$$

The semantics of φ, *denoted by* $[\![\varphi]\!]$, *is defined as the set* $\{w \in (2^P)^\omega \mid w \vDash \varphi\}$.

Definition 3 (Büchi Automata). *A nondeterministic Büchi automaton (NBA) over input alphabet* Σ *is a tuple* (Q, δ, I, F) *where* Q *is a finite set of states;* $\delta \subseteq Q \times \Sigma \times Q$ *is a set of transitions;* $I \subseteq \delta$ *is a set of initial transitions* [2] *and* F *is a set of final states.*

A run of a NBA over a word $w \in \Sigma^\omega$ *is an infinite sequence of states* $q_0 q_1 q_2 \dots$ *such that* $(q_0, w[0], q_1) \in I$ *and* $\forall\, i \geq 0$ $(q_i, w[i], q_{i+1}) \in \delta$. *A run is accepting if* $q_i \in F$ *for infinitely many* i.

The language accepted by an NBA $\mathcal{A}$, *denoted by* $L(\mathcal{A})$ *is the set of all words* $w \in \Sigma^\omega$ *which have an accepting run on* $\mathcal{A}$.

[2] We use initial transitions instead of states for notational convenience. It can be easily converted into a state based definition.

Definition 4 (Limit Determinism). *A NBA (Q, δ, I, F) over input alphabet Σ is said to be* limit deterministic *if for every state q reachable from a final state, it is the case that $|\delta(q, \sigma)| = 1$ for every $\sigma \in \Sigma$.*

Definition 5 (Mealy Automata). *A nondeterministic Mealy machine with Büchi acceptance (NBM) with input alphabet Σ and output alphabet Γ is a tuple (Q, δ, I, M, F) where (Q, δ, I, F) is an NBA with input alphabet Σ and $M : Q \times \Sigma \to \Gamma$ is a partial function that is defined on all (q, σ) for which there is a q' such that $(q, \sigma, q') \in \delta$.*

The relation accepted by an NBM is the set of all pairs $(w, \lambda) \in \Sigma^\omega \times \Gamma^\omega$ such that w is accepted by the NBA (Q, δ, I, F) and λ is such that there is an accepting run of w of the form $q_0 q_1 q_2 \ldots$ where $M(q_i, w[i]) = \lambda[i]$ for all i.

Definition 6 (Probabilistic Automata). *A probabilistic Büchi automaton (PBA) over input alphabet Σ is a tuple (Q, Δ, q_s, F) where Q is a finite set of states; $\Delta : Q \times \Sigma \times Q \to [0, 1]$ specifies transition probabilities such that for every $q \in Q$ and $\sigma \in \Sigma$ we have $\sum_{r \in Q} \Delta(q, \sigma, r) = 1$; $q_s \in Q$ is an initial state; $F \subseteq Q$ is a set of final states.*

Given a word $w \in \Sigma^\omega$ a PBA $\mathcal{M}$ behaves as follows: it is initially at state $q_0 = q_s$. After having seen the first i symbols $w[0]w[1] \ldots w[i-1]$ it is in state q_i. On seeing $w[i]$ it chooses the next state q_{i+1} with probability $\Delta(q_i, w[i], q_{i+1})$. It continues this process to produce a run $\rho \in Q^\omega$. A run ρ is accepting if some final state appears infinitely often.

A word w produces a Markov chain C obtained by unfolding the PBA $\mathcal{M}$ along the symbols of w [3]. The probability measure induced by this Markov chain on runs in Q^ω is used to define the acceptance probability of the word w on $\mathcal{M}$ as

$$\Pr(w) = \Pr\{\rho \in Q^\omega \mid \rho \text{ is accepting for } w \text{ over } \mathcal{M}\}$$

The language accepted by a PBA over input Σ denoted by $L_{>0}(\mathcal{M})$ is the set of all words $w \in \Sigma^\omega$ with positive acceptance probability, i.e $\Pr(w) > 0$.

Lemma 7. *[3] Given a limit deterministic NBA $\mathcal{D} = (Q, \delta, I, F)$ there is a PBA $\mathcal{M}$ with $|Q| + 1$ states such that $L(\mathcal{D}) = L_{>0}(\mathcal{M})$.*

Finally, we make note of the fact that any limit deterministic NBA translation for a fragment devoid of the $\boldsymbol{X}$ operator can be converted into translation for the fragment with $\boldsymbol{X}$ by incurring a multiplicative factor blow-up that is exponential in the number of nested $\boldsymbol{X}$s in the formula.

Proposition 8. *Let* LTL$'$ *be some fragment of* LTL*. If for every $\varphi \in$ LTL$'\backslash \boldsymbol{X}$ one can build a limit deterministic NBA $\mathcal{A}_\varphi$ such that it recognizes $[\![\varphi]\!]$ and is of size $f(|\varphi|)$, then for every $\varphi' \in$ LTL$'$ one can build a limit deterministic NBA $\mathcal{A}'_{\varphi'}$ such that it recognizes $[\![\varphi']\!]$ and has size $\mathcal{O}(2^n f(|\varphi'|))$ where n is the number of nested $\boldsymbol{X}$s appearing in φ'.*

Proof. Observe that $\boldsymbol{X}$ distributes over every temporal/boolean connective, and hence all the $\boldsymbol{X}$s in a formula can be pushed down to the level of propositions. Then we replace maximal formulae of the form $\boldsymbol{X}^k p$ with a fresh proposition and construct an automaton $\mathcal{A}$ for the formula devoid of $\boldsymbol{X}$s but over an extended set of propositions. This automaton can be augmented with an input buffer of length n which remembers n input symbols before starting to process $\mathcal{A}$. The extra propositions can then be projected out by looking at this buffer. □

3 Automata for LTL($\boldsymbol{F}$,$\boldsymbol{G}$) formulae

In this section, we present a construction of exponential sized limit deterministic NBA for the fragment LTL(F,G). Thanks to Proposition 8 we ignore the $\boldsymbol{X}$ operator since we are aiming to construct exponential size automata.

The following proposition embodies the key idea behind our construction.

Proposition 9. *For any formula $\varphi \in$ LTL over P, and any word $w \in (2^P)^\omega$ exactly one of the following three holds*

$$w \vDash \neg \boldsymbol{F}\varphi, \quad w \vDash (\neg \boldsymbol{G}\varphi \wedge \boldsymbol{F}\varphi), \quad w \vDash \boldsymbol{G}\varphi$$

Furthermore, if φ is of the form $\boldsymbol{F}\psi$ or $\boldsymbol{G}\psi$ then we can deduce if $w \vDash \varphi$ holds from knowing which one of the above three holds.

Proof. It is easy to see using propositional calculus that at least one of the three formula should hold, i.e $\neg \boldsymbol{F}\varphi \vee (\neg \boldsymbol{G}\varphi \wedge \boldsymbol{F}\varphi) \vee \boldsymbol{G}\varphi \equiv$ true. Conjunction of every pair of them is false. This can be proved using propositional calculus for $\neg \boldsymbol{F}\varphi \wedge (\neg \boldsymbol{G}\varphi \wedge \boldsymbol{F}\varphi)$, and for $(\neg \boldsymbol{G}\varphi \wedge \boldsymbol{F}\varphi) \wedge \boldsymbol{G}\varphi$. It is easy see that $\neg \boldsymbol{F}\varphi \wedge \boldsymbol{G}\varphi$ is false since $\neg \boldsymbol{F}\varphi$ says φ can never hold, while $\boldsymbol{G}\varphi$ asserts that φ is always true, which is a contradiction.

If φ is of the form $\boldsymbol{F}\psi$ then $w \vDash \varphi$ iff $w \nvDash \neg \boldsymbol{F}\varphi$. If φ is of the form $\boldsymbol{G}\psi$ then $w \vDash \varphi$ iff $w \vDash \boldsymbol{G}\varphi$. □

The essence of our construction is in guessing which one of the three formulae in Proposition 9 holds for $\boldsymbol{F}$ and $\boldsymbol{G}$ subformulae. We define a *guess*, to be tripartition $\pi = \langle \pi^\tau, \pi^\upsilon, \pi^\kappa \rangle$ of $\mathbb{F}_\varphi \cup \mathbb{G}_\varphi$. Let $\mathit{\Pi}$ denote the set of all tripartitions π of $\mathbb{F}_\varphi \cup \mathbb{G}_\varphi$. If a subformula $\boldsymbol{F}\psi$ is present in π^τ, our guess is that $\neg \boldsymbol{F}\psi (\equiv \neg \boldsymbol{F}\boldsymbol{F}\psi)$ holds for the input to be seen. If a subformula $\boldsymbol{G}\psi$ is in π^τ, our guess is that $\boldsymbol{G}\psi (\equiv \boldsymbol{G}\boldsymbol{G}\psi)$ holds for the input to be seen. Table 1 summarizes how we interpret a tripartition as one of three guesses for $\boldsymbol{F}$ and $\boldsymbol{G}$ subformulae.

From the second part of Proposition 9 we know exactly which formulae in $\mathbb{F}_\varphi \cup \mathbb{G}_\varphi$ are presently true according to $\pi \in \mathit{\Pi}$. This information along with the knowledge of the truth of propositions at present allows us to *evaluate* the truth of the formula φ according to π. We define what it means to evaluate a formula.

Definition 10 (Evaluation). *For any formula $\varphi \in$ LTL(F,G) over P, a partition $\pi \in \mathit{\Pi}$ and assignment ν on P we inductively define a boolean function $[\varphi]^\pi_\nu$, the evaluation of a formula φ, as follows:*

	π^τ	π^υ	π^κ
$\boldsymbol{F}\psi$	$\neg\boldsymbol{F}\psi$	$\boldsymbol{F}\psi \wedge \neg\boldsymbol{G}\boldsymbol{F}\psi$	$\boldsymbol{G}\boldsymbol{F}\psi$
$\boldsymbol{G}\psi$	$\boldsymbol{G}\psi$	$\neg\boldsymbol{G}\psi \wedge \boldsymbol{F}\boldsymbol{G}\psi$	$\neg\boldsymbol{F}\boldsymbol{G}\psi$

Table 1. Guess corresponding to a tripartition π

$$[\,p\,]^\pi_\nu = \nu(p) \qquad [\,\varphi \wedge \psi\,]^\pi_\nu = [\,\varphi\,]^\pi_\nu \wedge [\,\psi\,]^\pi_\nu \qquad [\,\boldsymbol{G}\psi\,]^\pi_\nu = \mathit{true}\ \mathit{iff}\ \boldsymbol{G}\psi \in \pi^\tau$$
$$[\,\neg p\,]^\pi_\nu = \neg\nu(p) \qquad [\,\varphi \vee \psi\,]^\pi_\nu = [\,\varphi\,]^\pi_\nu \vee [\,\psi\,]^\pi_\nu \qquad [\,\boldsymbol{F}\psi\,]^\pi_\nu = \mathit{true}\ \mathit{iff}\ \boldsymbol{F}\psi \notin \pi^\tau$$

Next, we observe that if π is sound in the sense that every formula $\boldsymbol{G}\psi \in \pi^\tau$ and $\boldsymbol{F}\psi \notin \pi^\tau$ is true at present then φ evaluates to true with respect to π indicates that φ is indeed true.

Proposition 11 (Soundness). *For any formula $\varphi \in \mathrm{LTL}(F,G)$, a guess $\pi \in \Pi$ and word $w \in \Sigma^\omega$ if the following implications hold*

$$\boldsymbol{G}\psi \in \pi^\tau \implies w \vDash \boldsymbol{G}\psi \qquad \boldsymbol{F}\psi \notin \pi^\tau \implies w \vDash \boldsymbol{F}\psi$$

for every $\boldsymbol{G}\psi$, $\boldsymbol{F}\psi$ that is a subformula of φ then: $[\,\varphi\,]^\pi_{w[0]} = \mathrm{true}$ implies $w \vDash \varphi$.

Proof. induction on the structure of φ. □

Similarly, if π is complete in the sense that every $\boldsymbol{G}\psi$ that is true is present in π^τ and every $\boldsymbol{F}\psi$ that is true is not in π^τ then φ is true implies it also evaluates to true.

Proposition 12 (Completeness). *For any formula $\varphi \in \mathrm{LTL}(F,G)$, a guess $\pi \in \Pi$ and a word $w \in \Sigma^\omega$ if the following implications holds*

$$w \vDash \boldsymbol{G}\psi \implies \boldsymbol{G}\psi \in \pi^\tau \qquad w \vDash \boldsymbol{F}\psi \implies \boldsymbol{F}\psi \notin \pi^\tau$$

for every $\boldsymbol{G}\psi$, $\boldsymbol{F}\psi$ that is a subformula of φ then: $w \vDash \varphi$ implies $[\,\varphi\,]^\pi_{w[0]} = \mathit{true}$.

Proof. induction on the structure of φ. □

In our construction, every state of the automaton holds a tripartition that corresponds to a guess. According to this guess some subformulae may hold now, in the future, or never. The initial guess is such that it enables the main formula to be true, and the subsequent guesses propagate temporal requirements. In an accepting run we ensure that each of our guesses is sound.

Since our formulae are in negation normal form, it only makes sense to take care of checking a guess on a subformula φ if the guess claims φ to be true at present or some point in the future. If a guess claims that φ does not hold we don't need to bother checking it, because even if it did it could not make a superformula false that was otherwise true. Therefore for $\boldsymbol{F}\psi$ we have to verify

the guess if it is present in π^υ or π^κ, and for $\boldsymbol{G}\psi$ we have to verify whenever it is present in π^τ or π^υ.

Say $\boldsymbol{G}\psi \in \pi^\tau$, it requires that $\boldsymbol{G}\psi$ holds now. This can be checked by ensuring ψ holds now and $\boldsymbol{G}\psi$ holds at the next time step. We can check if ψ is true at present by evaluating ψ with respect to our guess and the incoming input symbol. We can ensure ψ holds at the next time step by propagating $\boldsymbol{G}\psi$ to the π^τ in the next guess. If $\boldsymbol{G}\psi \in \pi^\upsilon$, we are guessing that $\neg\boldsymbol{G}\psi \wedge \boldsymbol{F}\boldsymbol{G}\psi$ holds. As observed earlier we need not ensure $\neg\boldsymbol{G}\psi$ holds, but we need to check that $\boldsymbol{F}\boldsymbol{G}\psi$ holds by having $\boldsymbol{G}\psi$ either in π^τ or π^υ of the next guess. But if we keep delaying $\boldsymbol{F}\boldsymbol{G}\psi$ by retaining $\boldsymbol{G}\psi$ in π^υ forever then $\boldsymbol{F}\boldsymbol{G}\psi$ will not hold. We overcome this by imposing an acceptance condition which requires the set π^υ to eventually become empty.

For $\boldsymbol{F}\psi \in \pi^\upsilon$ we need $\boldsymbol{F}\psi \wedge \neg\boldsymbol{G}\boldsymbol{F}\psi$ to hold. Once again we don't bother to check $\neg\boldsymbol{G}\boldsymbol{F}\psi$. For $\boldsymbol{F}\psi$ we need ψ to hold now or some point in the future. The former can be checked by evaluating ψ using the current guess and the incoming input symbol. If it does not evaluate to true we require $\boldsymbol{F}\psi$ to hold at the next time step, which is ensured by having $\boldsymbol{F}\psi$ in π^υ in the next step. Once again, this creates a possibility for $\boldsymbol{F}\psi$ to be false if it keeps getting delayed by its presence in π^υ forever, but as mentioned above our acceptance condition will be such that π^υ is eventually empty. For $\boldsymbol{F}\psi \in \pi^\kappa$ we are claiming $\boldsymbol{G}\boldsymbol{F}\psi$, this can be ensured by evaluating ψ with the respect to the current guess and input symbol, and requiring the resulting valuation to be true infinitely often. This can be achieved by fixing $\boldsymbol{F}\psi$ to be in π^κ forever and using an appropriate Büchi condition to verify that for $\boldsymbol{F}\psi$ in π^κ, ψ holds infinitely often.

Next we give our construction for an arbitrary LTL(F,G) formula.

Definition 13 (Construction). *Given a formula φ in* LTL(F,G) *defined over propositions P, let $\mathcal{D}(\varphi)$ be the NBA (Q, I, δ, F) over the alphabet 2^P defined as follows*

- *Q is the set $\Pi \times [z]$, consisting of guess-counter pairs where $z = |\mathbb{F}_\varphi|+1$*
- *δ is the set of all transitions*

$$(\pi_1, m) \xrightarrow{\nu} (\pi_2, n)$$

such that

(A) for each $\boldsymbol{G}\psi \in \pi_1^\tau$, $[\,\psi\,]_\nu^{\pi_1}$ is true

(B) for each $\boldsymbol{F}\psi \in \pi_1^\upsilon$, $[\,\psi\,]_\nu^{\pi_1}$ is false implies $\boldsymbol{F}\psi \in \pi_2^\upsilon$.

(C) $\pi_1^\tau \subseteq \pi_2^\tau$ and $\pi_1^\kappa = \pi_2^\kappa$

(D) n is updated as follows

$$n = \begin{cases} m, & (|\pi_1^\upsilon| > 0) \vee (m > 0 \wedge \neg[\,\psi_m\,]_\nu^{\pi_1}) \\ m{+}1 \pmod k, & \text{otherwise} \end{cases}$$

where $k = |\pi^\kappa \cap \mathbb{F}_\varphi| + 1$ and ψ_m be such that $\boldsymbol{F}(\psi_m)$ is the m^{th} formula in $\pi^\kappa \cap \mathbb{F}_\varphi$

– *I is the set of transitions of the form $(\pi, 0) \xrightarrow{\nu} (\pi', i)$ where $[\varphi]^{\pi}_{\nu}$ is true*
– *F is the set of states $(\pi, 0)$ where π^{υ} is empty.*

Next, we present the theorem that states the correctness of the above construction.

Theorem 14. *For any formula $\varphi \in \mathrm{LTL}(F,G)$, the NBA $\mathcal{D}(\varphi)$ is a limit deterministic automaton of size $2^{\mathcal{O}(|\varphi|)}$ such that $L(\mathcal{D}(\varphi)) = [\![\varphi]\!]$.*

Proof. The number of states in $\mathcal{D}(\varphi)$ is bounded by $3^{|\mathbb{F} \cup \mathbb{G}|} \times \log_2 |\mathbb{F}|$ and so clearly the size of $\mathcal{D}(\varphi)$ is exponential in $|\varphi|$.

We can see that $\mathcal{D}(\varphi)$ is limit deterministic as follows: The final states are of the form $(\pi, 0)$ where π^{υ} is empty. Note that according to condition (C), π^{υ} remains empty once it becomes empty, and π^{τ} and π^{κ} remain fixed. Hence the guess π can never change after visiting a final state. And since the counter is updated deterministically we have that any state reachable from a final state chooses its next state deterministically.

The proof of the fact $L(\mathcal{D}(\varphi)) = [\![\varphi]\!]$ is provided in the Appendix. □

4 Automata for LTL$\backslash GU$ formulae

In this section, we present a construction of limit deterministic NBAs for the fragment LTL$\backslash GU$ of exponential size. We follow a compositional approach where we compose a *master* and a *slave* automata (terminology borrowed from [6]) to obtain the required one. The master automaton assumes that the truth values of the maximal until-free subformulae are known at each step and checks whether the top-level until formula holds true. The master automaton works over an extended set of propositions where the new propositions are introduced in place of the until-free subformulae. The slave automaton works over the original set of propositions and outputs at each step the truth value of the subformulae abstracted by the master in the form of the new propositions. The master and the slave are then composed such that they work together to check the entire formula. Once again we apply Proposition 8 to ignore $\boldsymbol{X}$ operators when presenting our construction.

We first clarify some notation we use in this section. For a finite set of formulae Φ we use P_Φ to denote a set of propositions p_ϕ indexed by formulae $\phi \in \Phi$. We will use $\varphi_{/\Phi}$ to denote the formula obtained from φ by replacing subformulae of φ appearing in Φ by their corresponding propositions in P_Φ.

Definition 15 (Characteristic Relation). *For a finite set of* LTL *formulae Φ over propositions P we define its* characteristic relation $R_\Phi \subseteq (2^P)^\omega \times (2^{P_\Phi})^\omega$ *as follows:*

$$(w, \lambda) \in R_\Phi \quad \textit{iff} \quad \lambda[i] = \{p_\varphi \mid \varphi \in \Phi,\ w_i \vDash \varphi\}$$

Given $w_1 \in 2^{P_1}$ and $w_2 \in 2^{P_2}$ define the join of w_1 and w_2 denoted by $w_1 \cup w_2$ as the word in $(2^{P_1 \cup P_2})^\omega$ whose i^{th} element $(w_1 \cup w_2)[i]$ is $w_1[i] \cup w_2[i]$. Given a

relation $R \subseteq (2^{P_1})^\omega \times (2^{P_2})^\omega$ define the language $R^\circ \subseteq (2^{P_1 \cup P_2})^\omega$ as the set of words obtained by joining the pairs in R:

$$R^\circ = \left\{ \rho \in (2^{P_1 \cup P_2})^\omega \mid \exists (w, \lambda) \in R, \;\; \rho = w \cup \lambda \right\}$$

Later on in this section (Proposition 22) we will see how to construct a NBM for a set of LTL(F,G) formula which accepts a relation that is "subsumed" by the characteristic relation of the set. In that direction we define what it means for a relation to be subsumed by another.

Definition 16 (Subsumption). *For two relations $R, S \subseteq (2^{P_1})^\omega \times (2^{P_2})^\omega$ we say that R is* subsumed *by S, denoted by $R \triangleleft S$ iff $S \subseteq R$ and for every $(w, \lambda) \in R$ there exists $(w, \lambda') \in S$ such that $\forall i \; \lambda[i] \subseteq \lambda'[i]$.*

Next, we describe how to break an LTL$\backslash GU$ formula into an until factor and an until-free factor which are going to be handled by the master and slave automata respectively.

Definition 17 (Factorization). *Given a formula $\varphi \in$ LTL$\backslash GU$ over propositions P we identify the set of maximal subformulae of φ that do not contain $\boldsymbol{U}$ as the* until-free factor *of φ denoted by $\Upsilon(\varphi)$.*

$$\Upsilon(\boldsymbol{G}\varphi) = \{\boldsymbol{G}\varphi\} \qquad \Upsilon(\boldsymbol{F}\varphi) = \{\boldsymbol{F}\varphi\}$$

$$\Upsilon(\varphi_1 \wedge/\vee \varphi_2) = \begin{cases} \{\varphi_1 \wedge/\vee \varphi_2\} & \text{if } \varphi_1 \in \Upsilon(\varphi_1) \text{ and } \varphi_2 \in \Upsilon(\varphi_2) \\ \Upsilon(\varphi_1) \cup \Upsilon(\varphi_2) & \text{otherwise} \end{cases}$$

$$\Upsilon(\varphi_1 \boldsymbol{U} \varphi_2) = \Upsilon(\varphi_1) \cup \Upsilon(\varphi_2) \qquad \Upsilon(\ell) = \ell \quad \text{for literals } \ell$$

For a formula $\varphi \in$ LTL$\backslash GU$ we define its until factor *as the formula $\varphi_{/\Upsilon(\varphi)} \in$ LTL(U) simply written as χ_φ.*

The following proposition relates the semantics of an LTL$\backslash GU$ formula with the semantics of its until and until-free factors.

Proposition 18. *For any $\varphi \in$ LTL$\backslash GU$ over propositions P and any relation $R \subseteq (2^P)^\omega \times (2^{P_{\Upsilon(\varphi)}})^\omega$ such that $R \triangleleft R_{\Upsilon(\varphi)}$ we have*

$$[\![\varphi]\!] = (R^\circ \cap [\![\chi_\varphi]\!]) \restriction_{2^P}$$

Proof. We prove our statement by proving the following equivalence

$$w \vDash \varphi \iff (w \cup \lambda(w)) \vDash \chi_\varphi \tag{1}$$

where $\lambda(w) \in (2^{P_{\Upsilon(\varphi)}})^\omega$ is the unique word such that $(w, \lambda(w)) \in R_{\Upsilon(\varphi)}$. We do so by performing induction on φ:

i. $\varphi \in \Upsilon(\varphi)$: in which case $\chi_\varphi = p_\varphi$.

$$\begin{aligned} & w \vDash \varphi \\ \iff \quad & p_\varphi \in \lambda(w)[0] && \text{(definition of } \lambda(w)\text{)} \\ \iff \quad & (w \cup \lambda(w)) \vDash p_\varphi \end{aligned}$$

ii. $\varphi = (\varphi_1 \wedge/\vee \varphi_2) \notin \Upsilon(\varphi_1 \wedge/\vee \varphi_2)$: here $\chi_{\varphi_1 \wedge/\vee \varphi_2} = \chi_{\varphi_1} \wedge/\vee \chi_{\varphi_2}$

$$\begin{aligned}
& w \vDash \varphi_1 \wedge/\vee \varphi_2 \\
\iff \quad & w \vDash \varphi_1 \quad \text{and/or} \quad w \vDash \varphi_2 \\
\iff \quad & (w \cup \lambda(w)) \vDash \chi_{\varphi_1} \quad \text{and/or} \quad (w \cup \lambda(w)) \vDash \chi_{\varphi_2} \quad \text{(inductive hypothesis)} \\
\iff \quad & (w \cup \lambda(w)) \vDash (\chi_{\varphi_1} \wedge/\vee \chi_{\varphi_1}) = \chi_{\varphi_1 \wedge/\vee \varphi_2}
\end{aligned}$$

iii. $\varphi = (\varphi_1 \boldsymbol{U} \varphi_2)$: here $\chi_{\varphi_1 \boldsymbol{U} \varphi_2} = \chi_{\varphi_1} \boldsymbol{U} \chi_{\varphi_2}$

$$\begin{aligned}
& w \vDash \varphi_1 \boldsymbol{U} \varphi_2 \\
\iff \quad & \exists\, i \;\; w_i \vDash \varphi_2 \quad \text{and} \quad \forall\, j{<}i \;\; w_j \vDash \varphi_1 \\
\iff \quad & \exists\, i \;\; (w_i \cup \lambda(w_i)) \vDash \chi_{\varphi_2} \quad \text{and} \quad \forall\, j{<}i \;\; (w_j \cup \lambda(w_j)) \vDash \chi_{\varphi_1} \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(inductive hypothesis)} \\
\iff \quad & (w \cup \lambda(w)) \vDash \chi_{\varphi_1} \boldsymbol{U} \chi_{\varphi_2} = \chi_{\varphi_1 \boldsymbol{U} \varphi_2}
\end{aligned}$$

Now we also have $(w \cup \lambda(w)) \in R^\circ$ due to the fact that $(w, \lambda(w)) \in R_{\Upsilon(\varphi)}$ and $R_{\Upsilon(\varphi)} \subseteq R$. This along with (1) gives us $[\![\varphi]\!] \subseteq (R^\circ \cap [\![\chi_\varphi]\!]) \restriction_{2^P}$.

Next we observe that if $\lambda_1 \subseteq \lambda_2$ then $w \cup \lambda_1 \vDash \chi_\varphi$ implies $w \cup \lambda_2 \vDash \chi_\varphi$ because the propositions in $P_{\Upsilon(\varphi)}$ appear positively in χ_φ. Consider $w \in (R^\circ \cap [\![\chi_\varphi]\!]) \restriction_{2^P}$, this implies there is a λ such that $(w, \lambda) \in R$ and $(w \cup \lambda) \vDash \chi_\varphi$. Since $R \triangleleft R_{\Upsilon(\varphi)}$ we have $\lambda \subseteq \lambda(w)$ where $(w, \lambda(w)) \in R_{\Upsilon(\varphi)} \subseteq R$. Now from our first observation in this paragraph we have that $(w \cup \lambda(w)) \vDash \chi_\varphi$, from which we can conclude $w \vDash \varphi$ using (1). This proves the other side of the containment $(R^\circ \cap [\![\chi_\varphi]\!]) \restriction_{2^P} \subseteq [\![\varphi]\!]$. □

Let P_1 and P_2 be disjoint set of propositions. Let $\Sigma = 2^{P_1}$, $\Gamma = 2^{P_2}$ and by abusing notation let $\Sigma \times \Gamma = 2^{P_1 \cup P_2}$. Next we describe how to compose a master NBA $\mathcal{A}$ over input $\Sigma \times \Gamma$ and a slave NBM $\mathcal{B}$ over input alphabet Σ and output alphabet Γ to obtain an NBA $\mathcal{A} \times \mathcal{B}$ over input Σ.

Definition 19 (Composition). *Consider a NBA $\mathcal{A} = (Q_\mathcal{A}, \delta_\mathcal{A}, I_\mathcal{A}, F_\mathcal{A})$ over input alphabet $\Sigma \times \Gamma$ and a NBM $\mathcal{B} = (Q_B, \delta_\mathcal{B}, I_\mathcal{B}, M_\mathcal{B}, F_\mathcal{B})$ over input alphabet Σ and output alphabet Γ. We define a NBA $(Q_\mathcal{A} \times Q_\mathcal{B}, \delta_{\mathcal{A}\times\mathcal{B}}, I_{\mathcal{A}\times\mathcal{B}}, F_\mathcal{A} \times F_\mathcal{B})$ over input Σ denoted by $\mathcal{A} \times \mathcal{B}$ where*

$$\delta_{\mathcal{A}\times\mathcal{B}}((a_1, b_1), \sigma) = \{(a_2, b_2) \mid q_2 \in \delta_\mathcal{B}(q_1, \sigma),\ a_2 \in \delta_\mathcal{A}(a_1, \sigma \cup M_\mathcal{B}(b_1, \sigma))\}$$

$$I_{\mathcal{A}\times\mathcal{B}} = \{(a_1, b_1) \xrightarrow{\nu} (a_2, b_2) \mid b_1 \xrightarrow{\nu} b_2 \in I_\mathcal{B},\ a_1 \xrightarrow{\nu \cup M_\mathcal{B}(b_1, \nu)} a_2 \in I_\mathcal{A}\}$$

The proposition below relates the languages accepted by a master and a slave NBA to the language accepted by the product defined above. It requires the master to have final states such that on entering a final state it can never leave the set of final states. We shall refer to this property as absorbing final states.

Proposition 20. *Given a NBA $\mathcal{A}$ with input alphabet $\Sigma \times \Gamma$ with absorbing final states, and a NBA $\mathcal{B}$ with input alphabet Σ and output alphabet Γ we have*

$$L(\mathcal{A} \times \mathcal{B}) = (R_\mathcal{B}{}^\circ \cap L(\mathcal{A})) \restriction_\Sigma$$

Proof. For any $w \in \Sigma^\omega$

$w \in L(\mathcal{A} \times \mathcal{B})$

$\iff \exists$ a run $(a_0, b_0) \xrightarrow{w[0]} (a_1, b_1) \xrightarrow{w[1]} \dots$ which is accepting for $\mathcal{A} \times \mathcal{B}$

$\iff \exists$ states a_j, b_j for $j \geq 0$ s.t $(a_{i+1}, b_{i+1}) \in \delta_{\mathcal{A}\times\mathcal{B}}((a_i, b_i), w[i])$ for all i
and $(a_i, b_i) \in F_\mathcal{A} \times F_\mathcal{B}$ for infinitely many i (note $F_\mathcal{A}$ is absorbing)
and $(a_0, b_0) \xrightarrow{w[0]} (a_1, b_1) \in I_{\mathcal{A}\times\mathcal{B}}$

$\iff \exists$ states a_j, b_j for $j \geq 0$ s.t $b_{i+1} \in \delta_\mathcal{B}(b_i, w[i])$
and $a_{i+1} \in \delta_\mathcal{A}(a_i, w[i] \cup M_\mathcal{B}(b_i, w[i]))$ for all i
and $a_i \in F_\mathcal{A}$ for infinitely many i and $b_i \in F_\mathcal{B}$ for infinitely many i
$b_0 \xrightarrow{w[0]} b_1 \in I_\mathcal{B}$ and $a_0 \xrightarrow{w[0]\cup M_\mathcal{B}(b_0, w[0])} a_1 \in I_\mathcal{A}$

$\iff \exists$ states a_j, b_j for $j \geq 0$ s.t
the run $b_0 \xrightarrow{w[0]} b_1 \xrightarrow{w[1]} \dots$ is accepting for $\mathcal{B}$,
and the run $a_0 \xrightarrow{w[0]\cup M_\mathcal{B}(b_0, w[0])} a_1 \xrightarrow{w[1]\cup M_\mathcal{B}(b_1, w[1])} \dots$ is accepting for $\mathcal{A}$,

$\iff \exists \lambda \in \Gamma^\omega$ s.t $(w[0], \lambda[0])(w[1], \lambda[1]) \cdots \in R_\mathcal{B}$
and $w \cup \lambda \in L(\mathcal{A})$

$\iff w \in (R_\mathcal{B}{}^\circ \cap L(\mathcal{A})) \restriction_\Sigma$ □

The following proposition shows that the composition of a master and slave is limit deterministic if both of them are limit deterministic. The proof is provided in the Appendix.

Proposition 21. *Given a limit deterministic NBA $\mathcal{A}$ over input alphabet $\Sigma \times \Gamma$, and a limit deterministic NBM $\mathcal{B}$ with input alphabet Σ and output alphabet Γ it is the case that $\mathcal{A} \times \mathcal{B}$ is also limit deterministic.*

The next proposition illustrates how to construct a Mealy machine which recognizes a relation which is subsumed by the characteristic relation of an until-free factor, thus constructing a slave automaton of exponential size.

Proposition 22. *For any finite set $\Phi \subset$ LTL(F,G) over propositions P there is a NBM $\mathcal{B}_\Phi$ with input over 2^P and output over 2^{P_Φ} such that $R_{\mathcal{B}_\Phi} \lhd R_\Phi$, $\mathcal{B}_\Phi$ is limit deterministic and of size $\mathcal{O}(2^{|\Phi|})$*

Proof. Consider the construction of Theorem 14 with the following modifications to construct $\mathcal{B}_\Phi$:

- let Π be set of all three way partition of $\mathbb{G}_\Phi \cup \mathbb{F}_\Phi$ instead of $\mathbb{G}_\varphi \cup \mathbb{F}_\varphi$
- every transition of the form $(\pi, 0) \xrightarrow{\nu} (\pi', m)$ is initial
- define $M_\mathcal{B}((\pi, n), \nu)$ as $\{\, p_\varphi \in P_\Phi \mid [\varphi]^\pi_\nu = \mathit{true} \,\}$

The proof of $R_{\mathcal{B}_\Phi} \lhd R_\Phi$ can be found in the Appendix. □

Next we observe that master automaton can be constructed using a standard approach of translating an alternating automaton for the until factor to an NBA.

Proposition 23. *[17] For any formula $\varphi \in \mathrm{LTL}(U)$ there is a NBA $\mathcal{A}_\varphi$ with a single absorbing final state such that $L(\mathcal{A}_\varphi) = [\![\varphi]\!]$ and is of size $\mathcal{O}(2^{|\varphi|})$.*

Proof. As observed in Lemma 2 in [8], the construction follows from Theorem 22 and Proposition 20 of [17]. □

Finally we combine all the results in this section to show that the composition of the master and slave for the until and the until free components is as desired.

Theorem 24. *For any formula $\varphi \in \mathrm{LTL}\backslash GU$ the NBA $\mathcal{A}_{\chi_\varphi} \times \mathcal{B}_{\Upsilon(\varphi)}$ recognizes $[\![\varphi]\!]$, is limit deterministic and is of size $2^{\mathcal{O}(|\varphi|)}$.*

Proof. First we look at the language recognized by $\mathcal{A}_{\chi_\varphi} \times \mathcal{B}_{\Upsilon(\varphi)}$:

$$
\begin{aligned}
& L(\mathcal{A}_{\chi_\varphi} \times \mathcal{B}_{\Upsilon(\varphi)}) \\
& = (R_{\mathcal{B}_{\Upsilon(\varphi)}}{}^{\circ} \cap L(\mathcal{A}_{\chi_\varphi})) \restriction_{2^P} && \text{(Proposition 20)} \\
& = (R_{\mathcal{B}_{\Upsilon(\varphi)}}{}^{\circ} \cap [\![\chi_\varphi]\!]) \restriction_{2^P} && \text{(Proposition 23)} \\
& = [\![\varphi]\!] && \text{(Proposition 18 \& 22)}
\end{aligned}
$$

We have $\mathcal{A}_{\chi_\varphi} \times \mathcal{B}_{\Upsilon(\varphi)}$ to be limit deterministic from Proposition 21. The automata $\mathcal{A}_{\chi_\varphi}$ and $\mathcal{B}_{\Upsilon(\varphi)}$ are both exponential in the size of φ as seen in Propositions 23 & 22, and so the product is also of size $2^{\mathcal{O}(|\varphi|)}$. □

5 Results and Applications

In this section, we summarize and reflect on some of the consequences of our results related to PBAs and model checking concurrent probabilistic programs.

5.1 PBAs for LTL

First, we observe that our construction of limit deterministic NBAs gives us an exponential upper-bound on the size of PBAs for $\mathrm{LTL}\backslash GU$.

Theorem 25. *For any formula $\varphi \in \mathrm{LTL}\backslash GU$ there is a PBA $\mathcal{M}$ such that $L_{>0}(\mathcal{M}) = [\![\varphi]\!]$ and is of size $2^{\mathcal{O}(|\varphi|)}$.*

Proof. Follows from Lemma 7 and Theorem 24. □

Next, we note that this upper-bound is the best one can achieve.

Proposition 26. *There exists a family of formulae $\varphi_n \in \mathrm{LTL}(G)$ of size n such that any PBAs recognizing them have size at least 2^n.*

Proof. Consider the formula $\boldsymbol{G}(p \iff \boldsymbol{X}^n p)$. The language accepted by it is the set $\{u^\omega \mid u \in \Sigma^n\}$ where $\Sigma = \{\emptyset, \{p\}\}$. For each $u \in \Sigma^n$ there needs to be at least one state q_u such that u can reach q_u from the initial state with non-zero probability and no other $v \in \Sigma^n$ can reach q_u from the initial state with non-zero probability, because the word uv^ω should not be accepted with positive probability whereas u^ω should be accepted. □

Next, we note that constructing deterministic automata can result in a double exponential blow-up.

Proposition 27. *[1] There exists a family of of formulae $\varphi_n \in \mathrm{LTL}(F)$ such that any deterministic automata that recognize them have size $2^{2^{\Omega(n)}}$*

5.2 Model Checking MDPs

MDPs are the prevalent models for concurrent probabilistic programs. Concurrency is modeled as nondeterministic states, where the transitions are chosen by a scheduler. We refer the reader to [5] for a complete definition of MDPs. The model checking problem can be formulated as follows.

Definition 28. *Given a MDP $\mathcal{N}$ and temporal logic formula φ, the model checking problem is to decide if there exists a scheduler u such that $\Pr_{\mathcal{N},u}([\![\varphi]\!]) > 0$.*

Our construction of limit deterministic automata for LTL$\backslash GU$ leads to an improved EXPTIME model checking algorithm for MDPs and formulae in this logic which is matched by an EXPTIME-hard lower bound.

Theorem 29. *The model checking problem for MDPs and formulae in* LTL$\backslash GU$ *is* EXPTIME*-complete*

Proof. Proposition 4.2.3 in [5] states one can decide the model checking problem of MDPs and limit deterministic NBAs by taking a product of the two and doing a linear time analysis of the resulting graph. Using this along with our result in Theorem 24 we obtain the required upper bound. Proof of the lower bound can be found in the Appendix. □

We contrast this with the complexity of model checking MDPs and full LTL.

Proposition 30. *[5] The model checking problem for MDPs and formulae in* LTL *is* 2EXPTIME*-complete.*

6 Conclusions

In this paper, we have presented a translation of formulae in LTL$\backslash GU$ to probabilistic automata that are provably exponentially smaller than deterministic automata for this logic. This also yields a new, improved exponential time algorithm for model checking MDPs and LTL$\backslash GU$ as compared to the previously

known double exponential time complexity for MDPs and full LTL. Our automata in addition to having better upper-bounds also have a well defined logical structure, which makes it amenable to several optimizations.

There are few questions that are still left open. While we have shown how to construct exponential sized probabilistic and limit deterministic automata for LTL$\setminus GU$, it still remains open whether we can construct equally small probabilistic or limit deterministic automata for full LTL. In [8] we prove that translating the safety fragment of LTL to weak probabilistic monitors (which are special PBAs) can result in a double exponential blow-up. This might indicate that it is unlikely one will be able to give exponential sized PBAs for full LTL.

As a part of future work we intend to implement our construction and compare it with results for deterministic automata, and also see if our new algorithm for model checking yields better performance in practice.

References

1. Rajeev Alur and Salvatore La Torre. Deterministic generators and games for ltl fragments. *ACM Trans. Comput. Logic*, 5(1):1–25, January 2004.
2. T. Babiak, F. Blahoudek, M. Kretínský, and J. Strejeck. Effective translation of LTL to deterministic Rabin automata: Beyond the (F,G)-fragment. In *ATVA*, pages 24–39, 2013.
3. Christel Baier and Marcus Größer. Recognizing omega-regular languages with probabilistic automata. In *LICS*, pages 137–146, 2005.
4. Rohit Chadha, A. Prasad Sistla, and Mahesh Viswanathan. On the expressiveness and complexity of randomization in finite state monitors. *J. ACM*, 56(5):26:1–26:44, August 2009.
5. Costas Courcoubetis and Mihalis Yannakakis. The complexity of probabilistic verification. *J. ACM*, 42(4):857–907, 1995.
6. Javier Esparza and Jan Kretínský. From LTL to deterministic automata: A safraless compositional approach. In *CAV*, pages 192–208, 2014.
7. P. Gastin and D. Oddoux. Fast LTL to büchi automata translation. In *CAV*, pages 53–65, 2001.
8. Dileep Kini and Mahesh Viswanathan. Probabilistic automata for safety LTL specifications. In *VMCAI 2014, Proceedings*, pages 118–136, 2014.
9. J. Klein and C. Baier. Experiments with deterministic ω-automata for formulas of linear temporal logic. *Theoretical Computer Science*.
10. J. Kretínský and J. Esparza. Deterministic automata for the (F,G)-fragment of LTL. In *CAV*, pages 7–22, 2012.
11. J. Kretínský and R. Ledesma-Garza. Rabinizer 2: Small deterministic automata for LTL\ GU. In *ATVA*, pages 446–450, 2013.
12. A. Morgenstern and K. Schneider. From LTL to symbolically represented deterministic automata. In *VMCAI*, pages 279–293, 2008.
13. N. Piterman, A. Pnueli, and Y. Sa'ar. Synthesis of reactive(1) designs. In *VMCAI*, pages 279–293, 2008.
14. A. Pnueli and A. Zaks. On the merits of temporal testers. In *25 Years of Model Checking*, pages 172–195, 2008.
15. F. Somenzi and R. Bloem. Efficient Büchi automata from LTL formulae. In *CAV*, pages 248–263, 2000.

16. M. Vardi, P. Wolper, and A. P. Sistla. Reasoning about infinite computation paths. In *FOCS*, 1983.
17. Moshe Y. Vardi. An automata-theoretic approach to linear temporal logic. In *Logics for Concurrency. LNCS, vol. 1043*, pages 238–266. Springer-Verlag, 1996.

A Proofs of Section 3

Theorem 14. *For any formula $\varphi \in \mathrm{LTL}(F,G)$, the NBA $\mathcal{D}(\varphi)$ is a limit deterministic automaton of size $2^{\mathcal{O}(|\varphi|)}$ such that $L(\mathcal{D}(\varphi)) = [\![\varphi]\!]$.*

Proof. We prove that $\mathcal{D}(\varphi)$ accepts the correct language:
▶ $w \in L(\mathcal{D}(\varphi))$ implies $w \vDash \varphi$: let w be accepted by $\mathcal{D}$ with

$$(\pi_0, n_0) \xrightarrow{w[0]} (\pi_1, n_1) \xrightarrow{w[1]} \dots \tag{2}$$

as an accepting run. For $\boldsymbol{F}$ and $\boldsymbol{G}$ subformulae of φ we prove the following statements by performing induction on the formulae:

$$\forall i \quad \boldsymbol{G}\psi \in \pi_i^{\tau} \quad \Longrightarrow \quad w_i \vDash \boldsymbol{G}\psi \tag{3}$$
$$\forall i \quad \boldsymbol{F}\psi \in \pi_i^{\upsilon} \quad \Longrightarrow \quad w_i \vDash \boldsymbol{F}\psi \tag{4}$$
$$\forall i \quad \boldsymbol{F}\psi \in \pi_i^{\kappa} \quad \Longrightarrow \quad w_i \vDash \boldsymbol{G}\boldsymbol{F}\psi \tag{5}$$

The base case involves proving it when ψ is entirely propositional. Proving the base cases is similar to proving the inductive steps hence we combine them for the respective implications and prove it as follows:

Implication (3) We first observe that for any j if $\boldsymbol{G}\psi \in \pi_j^{\tau}$ implies $w_j \vDash \psi$. This is because (A) would say $[\,\psi\,]_{w[j]}^{\pi_j}$ is true, and applying Proposition 11 to π_j and w_j would gives us $w_j \vDash \psi$. (When ψ is not propositional we use the induction hypothesis to fulfil the requisite implications in Proposition 11).

Next, we have that if $\boldsymbol{G}\psi \in \pi_i^{\tau}$ then $\boldsymbol{G}\psi \in \pi_j^{\tau}$ for all $j > i$ by condition (C). This along with the previous argument implies $w_j \vDash \psi$ holds for all $j \geq i$ thus proving (3).

Implication (4) consider $\boldsymbol{F}\psi \in \pi_i^{\upsilon}$. Suppose $w_i \nvDash \boldsymbol{F}\psi$ which says $\forall\, j \geq i : w_j \nvDash \psi$, then using Proposition 11 (and the inductive hypothesis in the inductive case) we obtain that $[\,\psi\,]_{w[j]}^{\pi_j}$ is false for all $j \geq i$. Next, using condition (B) we get that $\boldsymbol{F}\psi \in \pi_j^{\upsilon}$ for all $j \geq i$. This violates the Büchi condition which requires π^{υ} to become empty, contradicting that the given run is accepting. Hence our assumption that $w_i \nvDash \boldsymbol{F}\psi$ is incorrect.

Implication (5) for $\boldsymbol{F}\psi \in \pi^{\kappa}$ we need to show that $w_i \vDash \boldsymbol{G}\boldsymbol{F}\psi$. It is sufficient to show that ψ holds infinitely often on some suffix of w. Consider i large enough so that π_i^{υ} is empty, such an i does exist because the run we consider is accepting in which the set π^{υ} at some point becomes empty because it has to visit a final state and remains empty because π^{υ} cannot grow along any valid transition due to (C). The Büchi condition enforces that the counter is 0 infinitely often, but since π^{κ} is non-empty (due to $\boldsymbol{F}\psi$) from condition (D) we get that the counter is incremrented infinitely often. Whenever the counter gets incremented from the index pointing to $\boldsymbol{F}\psi$ we get that the evaluation of ψ is true. Therefore we have $[\,\psi\,]_{w[j]}^{\pi_j}$ is true for inifinitely many $j \geq i$. Now using Proposition 11 (and the

inductive hypothesis in the inductive step) we obtain that ψ holds true infinitely often along w_i.

Having proved (3), (4) and (5) we now apply Proposition 11 and the initial condition of the automaton to conclude that $w \vDash \varphi$.

▶ $w \vDash \varphi$ implies $w \in L(\mathcal{D}(\varphi))$: for an index i, a word w and a set of formulae Ψ we define the following subsets of Ψ:

$$\mathcal{T}^i(\Psi) = \{\psi \in \Psi \mid w_i \vDash \psi\} \qquad \mathcal{F}^i(\Psi) = \{\psi \in \Psi \mid w_i \nvDash \psi\}$$
$$\mathcal{T}^i_\exists(\Psi) = \{\psi \in \Psi \mid w_i \vDash \boldsymbol{F}\psi\} \qquad \mathcal{F}^i_\exists(\Psi) = \{\psi \in \Psi \mid w_i \nvDash \boldsymbol{F}\psi\}$$
$$\mathcal{T}^i_\forall(\Psi) = \{\psi \in \Psi \mid w_i \vDash \boldsymbol{G}\psi\} \qquad \mathcal{F}^i_\forall(\Psi) = \{\psi \in \Psi \mid w_i \nvDash \boldsymbol{G}\psi\}$$

For the suffix w_i of the input w we define the guess π_i as:

$$\pi_i^\tau = \mathcal{T}^i(\mathbb{G}_\varphi) \cup \mathcal{F}^i(\mathbb{F}_\varphi)$$
$$\pi_i^\kappa = \mathcal{F}^i_\exists(\mathbb{G}_\varphi) \cup \mathcal{T}^i_\forall(\mathbb{F}_\varphi)$$
$$\pi_i^\upsilon = \big(\mathcal{F}^i(\mathbb{G}_\varphi) \cap \mathcal{T}^i_\exists(\mathbb{G}_\varphi)\big) \cup \big(\mathcal{T}^i(\mathbb{F}_\varphi) \cap \mathcal{F}^i_\forall(\mathbb{F}_\varphi)\big)$$

First observe that the sets π_i^κ and π_{i+1}^κ are the same for all i, and hence we drop the subscript for π^κ. We propose the following run for w which we will prove is valid and accepting

$$(\pi_0, n_0) \xrightarrow{w[0]} (\pi_1, n_1) \xrightarrow{w[1]} \dots \tag{6}$$

where we define n_i inductively as follows, let n_0 be 0. Let $\boldsymbol{F}(\psi_j)$ denote the j^{th} member of $\mathbb{F}_\varphi \cap \pi^\kappa$ according to some fixed ordering. Let n_{i+1} updated as follows:

$$n_{i+1} = \begin{cases} n_i & (|\pi_i^\upsilon| > 0) \vee (m > 0 \wedge w_i \nvDash \psi_{n_i}) \\ n_i{+}1 \pmod k & \text{otherwise} \end{cases}$$

where $k = |\mathbb{F}_\varphi \cap \pi^\kappa| + 1$.

We begin by showing that each state (π_i, n_i) is well defined. The triple $\langle \pi_i^\tau, \pi_i^\upsilon, \pi^\kappa \rangle$ is a partition because any formula in $\mathbb{F}_\varphi$ or $\mathbb{G}_\varphi$ lies in exactly one of the sets $\mathcal{T}^i(\mathbb{G}_\varphi)$, $\mathcal{F}^i_\exists(\mathbb{G}_\varphi)$, $\mathcal{F}^i(\mathbb{G}_\varphi) \cap \mathcal{T}^i_\exists(\mathbb{G}_\varphi)$ as argued in Proposition 9.

Next we show that for each $i \geq 0$, the transition $(\pi_i, n_i) \xrightarrow{w[i]} (\pi_{i+1}, n_{i+1})$ is present in δ by going through the conditions (A) to (D).

Condition (A): If $\boldsymbol{G}\psi \in \pi_i^\tau$ then from definition of π_i^τ we have $w_i \vDash \boldsymbol{G}\psi$. In particular $w_i \vDash \psi$. Now using Proposition 12 on π_i and w_i we obtain that $[\,\psi\,]^{\pi_i}_{w[i]}$ is true thus satisfying condition (A).

Condition (B): For any $\boldsymbol{F}\psi \in \pi_i^\upsilon$ we have $w_i \vDash \boldsymbol{F}\psi$ due to definition of π_i^υ. This implies either $w_i \vDash \psi$ or $w_{i+1} \vDash \boldsymbol{F}\psi$ has to be true. If it is the former then using Proposition 12 on π_i and w_i gives us that $[\,\psi\,]^{\pi_i}_{w[i]}$ is true. In case of the latter using definition of π_{i+1}^υ we obtain that $\boldsymbol{F}\psi \in \pi_{i+1}^\upsilon$.

Condition (C): For any $\boldsymbol{F}\psi \in \pi_i^\tau$ we know $w_i \not\models \boldsymbol{F}\psi$ which implies ψ never holds along w_i, and therefore never holds along the suffix w_{i+1} and so we have $w_{i+1} \not\models \boldsymbol{F}\psi$, and hence $\boldsymbol{F}\psi \in \pi_{i+1}^\tau$. Similarly for $\boldsymbol{G}\psi \in \pi_i^\tau$ we know $w_i \models \boldsymbol{G}\psi$, and therefore $w_{i+1} \models \boldsymbol{G}\psi$ which then implies $\boldsymbol{G}\psi \in \pi_{i+1}^\tau$.

Condition (D): Using Proposition 11 & 12 on π_i and w_i we get that $[\,\psi\,]_{w[i]}^{\pi_i}$ is equivalent to $w_i \models \psi$ for any subformula ψ of φ. This gives us that n_{i+1} is updated exactly as required in Condition (D).

Initial Condition: Applying Proposition 12 on π_0 and w we get that $w \models \varphi$ implies $[\,\varphi\,]_{w[0]}^{\pi_0}$. Since w is assumed to satisfy φ we get that $[\,\varphi\,]_{w[0]}^{\pi_{i+1}}$ is true which confirms that $(\pi_0, n_0) \xrightarrow{w[0]} (\pi_1, n_1)$ is a valid initial transition.

Büchi Condition: For any $\boldsymbol{G}\psi \in \pi_i^\upsilon$ we know that $w_i \models \boldsymbol{F}\boldsymbol{G}\psi$, which implies that there is a j large enough such that $w_j \models \boldsymbol{G}\psi$ and therefore $\boldsymbol{G}\psi \in \pi_j^\tau$ and $\notin \pi_j^\upsilon$. For any $\boldsymbol{F}\psi \in \pi_i^\upsilon$ we have that $w_i \not\models \boldsymbol{G}\boldsymbol{F}\psi$. This means that ψ does not hold infinitely often and so for some large enough j we obtain that $w_j \not\models \boldsymbol{F}\psi$ and therefore $\boldsymbol{F}\psi \in \pi_j^\tau$ and $\notin \pi_j^\upsilon$. This implies that π_j^υ becomes empty for a sufficiently large j. All that remains to be shown is that counter is 0 infinitely often. If $\pi^\kappa \cap \mathbb{F}_\varphi$ is empty then the counter is always 0, so consider the case when it is non-empty. Now we prove that the counter n gets incremented infinitely often, thus attaining 0 infinitely often. If this is not the case the counter would be stuck at a non-zero value, say m for the entirety of some suffix w_i because the values taken by the counter if updated infinitely often cycle through $[k]$. Let the m^{th} formula in $\pi^\kappa \cap \mathbb{F}$ be $\boldsymbol{F}\psi$. From the definition of n_{i+1} we obtain that $w_j \not\models \psi$ for all sufficiently large j and hence $w \not\models \boldsymbol{G}\boldsymbol{F}\psi$. But from the definition of π^κ we get that $w \models \boldsymbol{G}\boldsymbol{F}\psi$, a contradiction and hence n is 0 infinitely often. □

B Proofs of Section 4

Proposition 21. *Given a limit deterministic NBA $\mathcal{A}$ over input alphabet $\Sigma \times \Gamma$, and a limit deterministic NBM $\mathcal{B}$ with input alphabet Σ and output alphabet Γ it is the case that $\mathcal{A} \times \mathcal{B}$ is also limit deterministic.*

Proof. Let (a, b) be a state reachable from some final state $(a_0, b_0) \in F_\mathcal{A} \times F_\mathcal{B}$. We have finite word $w \in \Sigma^*$ such that

$$(a_0, b_0) \xrightarrow{w[0]} (a_1, b_1) \xrightarrow{w[1]} \dots \xrightarrow{w[n]} (a, b)$$

This implies that $a_0 \xrightarrow{w[0] \cup M_\mathcal{B}(b_0, w[0])} a_1 \xrightarrow{w[1] \cup M_\mathcal{B}(b_1, w[1])} \dots \xrightarrow{w[n] \cup M_\mathcal{B}(b_n, w[n])} a$ is a valid sequence of transitions in $\mathcal{A}$ and $b_0 \xrightarrow{w[0]} b_1 \xrightarrow{w[1]} \dots \xrightarrow{w[n]} b$ is a valid sequence of transitions in $\mathcal{B}$. Now this makes a reachable from a_0 and b reachable from b_0 and using the fact that $\mathcal{A}$ and $\mathcal{B}$ are limit deterministic we obtain that (a, b) can only make deterministic choices. □

Proposition 22. *For any finite set $\Phi \subset \mathrm{LTL}(F,G)$ over propositions P there is a NBM $\mathcal{B}_\Phi$ with input over 2^P and output over 2^{P_Φ} such that $R_{\mathcal{B}_\Phi} \triangleleft R_\Phi$, $\mathcal{B}_\Phi$ is limit deterministic and of size $\mathcal{O}(2^{|\Phi|})$*

Proof. Consider any word $w \in \Sigma^\omega$ and let $(\pi_0, n_0), (\pi_1, n_1), \dots$ be the sequence of states as defined in the run (6) with $\mathbb{F}_\varphi$ and $\mathbb{G}_\varphi$ replaced with $\mathbb{F}_\Phi$ and $\mathbb{G}_\Phi$ respectively. It is easy to verify that the arguments proving the validity of the run (6) still goes through and hence w is accepted by $\mathcal{B}_\varphi$.

Now we prove $R_{\mathcal{B}_\Phi} \triangleleft R_\Phi$ as required by Definition. We first prove that $R_\Phi \subseteq R_{\mathcal{B}_\Phi}$:

$$
\begin{aligned}
&(w, \lambda) \in R_\Phi \\
&\implies \forall\, i \;\; \lambda[i] = \{\, p_\varphi \in P_\Phi \mid w_i \vDash \varphi\} \\
&\implies \forall\, i \;\; \lambda[i] = \{\, p_\varphi \in P_\Phi \mid [\,\varphi\,]_{w[i]}^{\pi_i} = \mathit{true}\} \quad \text{(applying Proposition 11 \& 12 on } \pi_i, w_i) \\
&\implies \forall\, i \;\; \lambda[i] = M_\mathcal{B}((\pi_i, n_i), w[i]) \\
&\implies (w, \lambda) \in R_{\mathcal{B}_\Phi}
\end{aligned}
$$

Next, for $(w, \lambda) \in R_{\mathcal{B}_\Phi}$ we prove $\lambda \subseteq \lambda(w)$, where $\lambda(w)$ is the unique word such that $(w, \lambda(w)) \in R_\Phi$. Consider any $w \in \Sigma^\omega$ such that w is accepted by $\mathcal{B}_\Phi$ and let $(\pi_0, n_0), (\pi_1, n_1), \dots$ be a sequence of states witnessing the acceptance as considered in (2). The implications (3),(4),(5) can once again shown to be true in the same way.

$$
\begin{aligned}
&(w, \lambda) \in R_{\mathcal{B}_\Phi} \\
&\implies \forall\, i \;\; \lambda[i] = M_\mathcal{B}((\pi_i, n_i), w[i]) \\
&\implies \forall\, i \;\; \lambda[i] = \{\, p_\varphi \in P_\Phi \mid [\,\varphi\,]_{w[i]}^{\pi_i} = \mathit{true}\,\} \\
&\implies \forall\, i \;\; \lambda[i] \subseteq \{\, p_\varphi \in P_\Phi \mid w_i \vDash \varphi\} \quad \text{(applying Proposition 11 on } \pi_i, w_i) \\
&\implies \forall\, i \;\; \lambda[i] \subseteq \lambda(w)[i]
\end{aligned}
$$

C Proofs of Section 5

Theorem 29. *The model checking problem for MDPs and formulae in* LTL$\backslash GU$ *is* EXPTIME*-complete*

Proof. We present a sketch of the EXPTIME-hard lower bound for this problem. We reduce the membership problem for linear space-bounded alternating Turing machine to our model checking problem along the lines of the reduction in Theorem 3.2.1 in [5]. Given such a TM T and an input x we construct a MDP $\mathcal{N}$ and $\varphi \in \mathrm{LTL}(F,G)$ both polynomial in the size of the T and x such that T accepts x iff there is a scheduler u that admits $\Pr_{\mathcal{N},u}([\![\varphi]\!]) > 0$. The computation of an alternating machine can be viewed as a two player game consisting of the existential player E and the universal player U where the objective of E is to reach an accept state of T and the objective of U is to reach a reject state. The input x is accepted by T iff E has a winning strategy. The MDP we construct will be

such that the scheduler will simulate E and the stochastic choices will simulate U. Since a stochastic player is not as capable as U we give it infinitely many attempts to defeat E who is being played by the intelligent scheduler. This is done by replaying a game on reaching an accept state and terminating the game on reaching a reject state. If E indeed has a winning strategy then there is a u which plays such that the reject state will never be reached and an accept state will be visited infinitely often. If U has a winning strategy then sooner or later the stochastic player will play the winning moves in order to reach a reject state, and hence no scheduler has a chance to visit a final state infinitely often with non-zero probability. Thus the input x is accepted by T iff there is a scheduler u such that it can visit the accept state infinitely often with probability > 0. The requirement that the scheduler visit the accept state infinitely often can be formulated as $\boldsymbol{GF}$ formula. The issue that remains is that the MDP cannot store the entire configuration of T in its state as there are exponentially many. In order to remedy this what the MDP does is remember the head position, symbol beneath the head and the current state. It then allows the scheduler to guess the complete configuration. In order to prevent the scheduler from cheating we impose restrictions on how it can choose the configurations using appropriate temporal formula. Next we summarize the working of the MDP $\mathcal{N}$

(1) $\mathcal{N}$ begins by allowing u to guess the inital configuration of T as a sequence of states, where at i^{th} state the contents of the i^{th} tape cell is guessed. Note that the entire configuration is not stored but traversed from left to right.
(2) $\mathcal{N}$ remembers the location of the head, symbol beneath the head and the state of T.
(3) depending upon whether the configuration is in a E or U state the scheduler or the stochastic player respectively is allowed to choose the next transition
(4) The scheduler then attempts to generate the next configuration of T as a sequence of states. $\mathcal{N}$ forces it to correctly guess the symbol to be written on the tape, update the position of the head and the state of T.
(5) If an accept state is reached repeat from Step 1. If a reject state is reached halt the computation. Otherwise continue the computation of T by moving to Step 2.

For the configuration generated by the scheduler in Step 4 to be correct we enforce the cell contents not under the head to remain the same by using the following formula

$$\bigwedge_{i=1}^{n} \boldsymbol{G}\big((\neg h_i \wedge c) \implies (p \iff \boldsymbol{X}^{n+1} p)\big)$$

where n is the length of the input x, p denotes the contents of the current cell (as generated by the scheduler in Steps 1 and 4), h_i denotes whether the head is over i^{th} cell , and c denotes that the next state is neither accept nor reject. In order to ensure that the initial configuration is correctly generated we have a proposition b_i that is true only when it is guessing the contents of the i^{th} cell in Step 1 and then enforce $\wedge_{i=1}^{n} \boldsymbol{G}(b_i \implies (p \iff x[i]))$. □